IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2024.0429000

A Distribution Agnostic Rank Based Measure for Proximity Search

MAYUR GARG[1], ASHUTOSH NAYAK[2], and RAJASEKHARA REDDY DUVVURU MUNI[2]
[1]United Airlines, Gurugram, India
[2]Samsung Research Institute Bangalore, India

Corresponding author: Rajasekhara Reddy Duvvuru Muni (e-mail: raja.duvvuru@samsung.com).

ABSTRACT Proximity search is extensively used in modern machine learning algorithms across various applications. Proximity search aims at finding data points which are close to the data point of interest. Extant algorithms depend on distance-based metrics to find the closest data points. However, these metrics are limited by their dependency on the distribution of data along different dimensions, making them sensitive to scaling and translation. The performance also suffers as the number of dimensions increase. Furthermore, proximity estimation between any two data points in extant metrics does not factor in the relative position of the rest of the data. In this paper, we aim to provide an alternative to these metrics by proposing Rank Adjacency Measure (*RAM*) which is agnostic to the distribution of the data. RAM estimates the probability of proximity between points by extending the concept of ordering in one dimension. We provide a detailed mathematical construction of *RAM*. We illustrate the effectiveness of the proposed methodology using five datasets in three application areas - Outlier Detection, Nearest Neighbor Search, and Text Similarity. While our proposed methodology outperforms existing algorithms in outlier detection by 50%, it performs at par with existing metrics for other two applications. We conclude the paper with discussion on its limitations and research directions for improving RAM.

INDEX TERMS Rank based Methods, Nearest Neighbor, Outlier Detection, Unsupervised Learning, Proximity Search, Text Similarity.

I. INTRODUCTION

Different applications of machine learning models touch every aspect of our digital lives. The market size of machine learning is estimated to be $ 26.03 Billion dollar in 2023, which is expected to grow to $ 225.91 Billion dollars by 2030 [1]. Most of these applications use some proximity metrics to identify data points/instances closer to the data point of interest. To find these proximal data points, there are several well-studied and widely used metrics like euclidean distances, cosine similarity. However, these metrics suffer from inherent disadvantages and complexities. With an objective to provide an alternative for improved proximity search, we introduce a novel rank-based proximity metric called *RAM* in this paper. We present a detailed theoretical model for *RAM* and discuss its applications and implementation.

Proximity search is extensively used in different machine learning applications. The idea is to find a data instance (data point or context vector) which is closest to the data instance of interest. For example, campaign managers want to find an existing consumer whose characteristics are similar to a new consumer so that they can send top performing campaigns to the new consumer. In this case, the campaign manager will look at consumers with similar user demographics and behavior history e.g. age, salary, number of clicks, purchases etc. Similarly, if a consumer purchases a toothbrush, machine learning algorithms (e.g. recommendation system) finds products e.g. toothpaste whose embeddings are closest to the embeddings of toothbrush. In the first example, if we have age, salary and past purchase amount of the consumers, simple distance-based proximity metrics will be driven only by the salary due to scale difference. Thus, to efficiently run this model, the campaign manager should have information about the distribution of different dimensions and henceforth perform scaling operations to improve the model performance. In the second example, if the model considers cosine similarity, the model may fail to differentiate toothpastes whose embeddings are closer to the context vector (in embedding space) as cosine similarity only considers direction but not the magnitude of difference between the vectors.

Distance based metrics require prior knowledge of the data distributions in different dimensions, which makes it prohibitive for various machine learning algorithms to effectively use these metrics. Also, some distance based metrics assume Gaussian distribution which is not common in practical applications [2]. These metrics also suffer from difference in the scale of data along different dimensions and may require an additional step of normalizing the data set before any useful implementation. Distance based metrics face collapsing as the number of dimensions increase. That is, the variation in the distances between data points decreases, making it very sensitive for finding proximal points. This further makes it difficult for the downstream usage of metrics e.g. identifying the nearest data instances within a distance threshold. *RAM* proposed in this paper uses ranking for proximity search and is agnostic to the data distribution and the dimensionality of the data.

Rank-based metric do not suffer from both the disadvantages discussed in the previous examples. Rank-based metrics inherit the advantages of rank based methods, that is robustness and non-requirement of distribution knowledge. Rank-based methods are robust to extreme high or low values for a data instance and they consider ranks and not the actual values. Rank-based methods sort the data instances based on rank, thus we do not need to know the underlying distribution of the data set along different dimensions. However, rule-based metrics suffer from other limitations as we discuss in detail in Section III-E. Extant literature on rank-based metrics/measures do not discuss aggregate ranking in multiple dimensions.

Motivated by the advantages presented by rank-based metrics and limitations in extant metrics, we propose a novel approach for proximity search namely *RAM*. The major contributions of this paper include:

- we present a novel approach for proximity search using rank-based metric namely *RAM*
- we provide a detailed mathematical construction, limitations and implementation details for *RAM*
- we evaluate the performance of the proposed measure on five different datasets across three test cases
- we introduce a novel approach for unsupervised outlier detection using rank-based methods in *RAM*

Rest of the paper is organized as: We present related extant literature in Section II. We follow it up with a detailed discussion in Section III on the theoretical and mathematical foundation for *RAM*. To validate and illustrate the effectiveness of the *RAM*, we test and report the performance of the model across three test cases using five different datasets in Section IV. We conclude the paper in Section V with discussion on future research directions.

II. RELATED WORK

Proximity search is extensively used in modern machine learning applications. It aims to find data points that are proximal (close) to the data point of interest (next item prediction in a basket) or context of interest (user current session in news recommendation [3]). Proximity search is extensively used in various machine learning applications e.g. context retrieval [4] or Deduplication [5]. Bhatia and Vandana [6] present a comprehensive survey of various nearest neighbor techniques. More recently, Large Language Models (LLMs) have popularized vector databases which finds text embeddings which are closest to the text embeddings of the conversational context to retrieve chat answers [7]. Vector databases also use proximity search to match conversational context to the answer set.

A. DISTANCE METRICS

Proximity is defined as the inverse of ‘‘distance" between two points/instances, that is, a data point is closest to the data point of interest if the distance between them is smaller as compared to other data points. Different existing methods on proximity search define this ‘‘distance" differently. Note that we use methods/metrics/measure interchangeably in this paper. There are multiple works to measure the distance between two points. Most commonly used metrics include Euclidean distance (spherical distance), cosine similarity (angular distance), correlation (linear dependence) or KL Divergence (intersection of probability distributions). A comprehensive summary of the different distance based measures is provided by Cha [8]. This paper provides analysis on the similarity between these measures. Their study using random probability density functions demonstrate that angular based similarity coefficients such as cosine, Jaccard, and Dice [9] are closely related to the Euclidean distance.

Although these metrics are commonly used, they inherit certain disadvantages. The extant metrics depend extensively on the density, distribution and dimensionality of the space they lie in. Gower [10] used range of the values to calculate distance for quantitative features. However, the use of range is very sensitive to the presence of outliers. Due to the curse of dimensionality as discussed by Bellman and Kalaba [11], all points may appear equidistant to each other in high dimensional spaces. Many of the extant metrics are also scale-invariant which makes them sensitive to transformation in scale.

We contribute to the literature on proximity measures by proposing *RAM* - Rank Adjacency Measure. *RAM* is based on ranking which we discuss next.

B. RANK-BASED METHODS

RAM proposed in this paper is based on ranking the data points across dimensions. Rank is a well studied field in literature and used in multiple applications including recommendation systems [12], classification [13] and information retrieval [14].

Rank based metrics first sort the data instances according to their values. Rank based methods are robust to the scale of the data and the presence of any outliers. Rank based methods do not assume any distribution of data. Rank based methods are well known in machine learning community(e.g. linear regression [15] or finding correlation [16]). However, there is

no intuitive understanding of ranking for higher order dimensions. Bagui et al [17] and Fagin et al [18] have attempted to aggregate ranking for multivariate data. Bagui et al [17] introduced a heuristic score function that uses the distribution parameters of each dimension to rank all data points for k-RNN classification. Although this scoring is non-parametric, it works better for Gaussian style distributions which may not be true for all dimensions. Fagin et al [18] has implemented a median rank aggregation strategy to rank nearest neighbors from each dimension for similarity search that is computationally faster than computing Euclidean distances to all points. We extend the idea of identifying proximal points by ranking them in each dimension. We propose a novel direction for aggregating the ranks by defining proximity as the probability of the points of interest being in the same subspace when the space is partitioned.

We add to the literature on rank based methods by introducing a new measure called *RAM* for proximity search. Furthermore, we explore a new application of rank based methods – outlier detection. Our experiments show that *RAM* outperforms existing methods significantly in outlier detection.

III. RANK ADJACENCY MEASURE (*RAM*)

In this section, we provide a detailed discussion on the proposed method for measuring proximity – Rank Adjacency Measure (RAM). First, we illustrate the idea of the proposed heuristic with an intuitive example and then we construct its mathematical foundation. Next, we discuss the properties of RAM for its practical implementations. Finally, we end the section by highlighting the limitations of the proposed heuristics.

A. INTUITION

An illustrative example for understanding the idea behind RAM is shown in Figure 1. The data set contains seven data points (plotted on 2-dimensional space S). A partitioning line can be drawn perpendicular to the *x-axis* in S (dotted lines) such that it divides the set of points into two sets - one on either side of the partitioning line. For seven data points, seven such lines are possible (we use any one of the two extreme partitioning lines because they result in the same set of partitions). Such partitioning lines can also be drawn perpendicular to the *y-axis* (dashed lines). All such possible partitioning lines for the given data points are shown in Figure 1. A random pair of partitioning lines (one from each dimension) partitions S into 4 quadrants as shown in Figure 2.

To find if any data point A is close to data point B, we divide the space into different quadrants using a pair of partitioning lines. The total combinations of quadrants in which S can be partitioned into is 7^2 since partitioning across one axis is independent of the other. For any data set of N points with D dimensions, this generalizes to N^D combinations. The proximity of point A to B can then be expressed as the ratio of the number of times both lie in the same quadrant to the total number of such partitions. In other words, the proximity

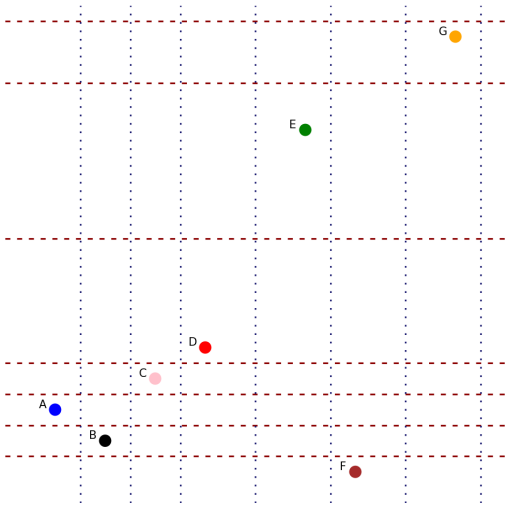

FIGURE 1: All possible partitions of a 2-Dimensional Space S into sub-spaces

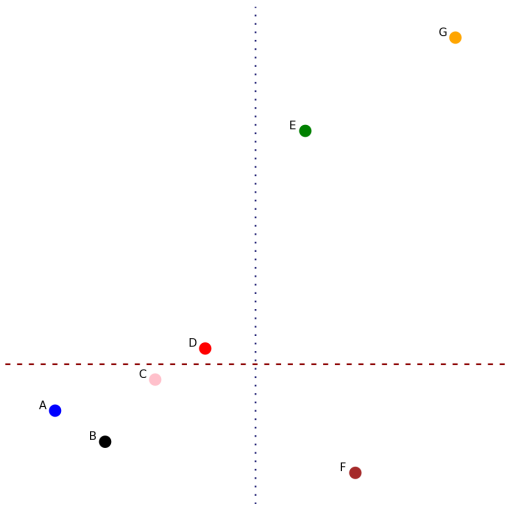

FIGURE 2: Example of a random partition using a pair of lines across each dimension

of point A and B is related to the probability of them lying in the same sub-space when the multi-dimensional space is randomly partitioned once along each axis by hyper planes.

The number of partitioning hyper planes perpendicular to any axis between two points is equal to the difference in their ranks when they are sorted along that axis. All other partitioning hyper planes perpendicular to that axis cannot separate those points into different half-space. Hence, along each axis, the probability of any two points lying in the same half-space when partitioned is proportional to the difference between the total number of points and the difference of rank of the two points of interest along an axis. Since partitioning along each axis is independent of the other, we multiply these probabilities for each axis for the same pair of points to calculate the overall probability of the data points lying in the

same sub-space.

Next, we discuss the mathematical construct of the proposed method in detail.

B. MATHEMATICAL CONSTRUCT

RAM aims to find the proximity $RAM_{I,i,j}$ between the data points $i, j \in I$ in space S where $|I| = N$ is the total number of data points. Let the data point i be represented by a D-dimensional vector $x_i \in R^D$. Let $H_d \in R^{D-1}$ be a $D-1$ dimensional hyperplane that partitions the space S into two sub-spaces perpendicular to dimension $d \in D$. For each dimension in D, the space can be partitioned in at most N partitions which includes 1 trivial partition. Let $Rank_d$ be a function in $R^D \rightarrow Z^+$ that finds the rank of any point i along the dimension d w.r.t. I.

The probability of two points i, j lying in the same sub-space when partitioned by a random H_d for each dimension $d \in D$ is shown in Equation 1.

$$Prob_{i,j,H_d} = \prod_{d \in D} \left(\frac{N - |Rank_d(i) - Rank_d(j)|}{N} \right) \quad (1)$$

Since the probability of two unique points lying in the same half when partitioned by H_d along dimension d is in the range $(0, 1)$, multiplying such probabilities for each dimension results in small values, especially as D increases. To counter this effect, we normalize $Prob_{i,j,H_d}$ as shown in Equation 1.

$$RAM_{I,i,j} = \sqrt[D]{\prod_{d \in D} \left(\frac{N - |Rank_d(i) - Rank_d(j)|}{N} \right)} \quad (2)$$

C. PROPERTIES OF RAM

In this section, we discuss the different properties of RAM.

1) Commutativity

RAM uses absolute difference of ranks as shown in Equation 2, hence RAM is commutative.

$$RAM_{I,i,j} = RAM_{I,j,i} \quad \forall i, j \in I \quad (3)$$

2) Injectivity when RAM = 1

For any point $i \in I$, the rank adjacency of i with itself can be defined using Equation 2 as:

$$RAM_{I,i,i} = \sqrt[D]{\prod_{d \in D} \left(\frac{N}{N} \right)} = 1 \quad (4)$$

Since every deterministic ranking function $Rank_d$ is injective i.e. $Rank_d(i) = Rank_d(j) \quad \forall d \in D \implies i = j$, we state that:

$$RAM_{I,i,j} = 1 \implies i = j \quad (5)$$

3) Invariant to Isotropic and Anisotropic Scaling

If all points in I are scaled by a real vector $M = \{m_1, m_2, \ldots m_d\}$ such that dimension d is scaled by a factor of $m_d \neq 0$, we can state that for all points $i \rightarrow i_M, j \rightarrow j_M \forall i, j \in I$:

$$RAM_{I_M,i_M,j_M} = RAM_{I,i,j} \quad (6)$$

This is because RAM depends on the absolute difference of ranks between the two data points and scaling any axis by a non-zero real factor doesn't change the relative ordering of the points. This is true for identical (isotropic) and different (anisotropic) scaling factors along each dimension. Using ranks allows us to ignore any change in the density of the space and only consider the proportion of points that lie between any two points. This property becomes important when standardization or normalization of data is required since existing proximity calculation methods such as Minkowski family of distances or cosine similarity are prone to anisotropic scaling.

4) Invariant to Origin Shift

If the origin of the space S undergoes translation based on a real vector $T = \{t_1, t_2, \ldots t_d\}$ such that dimension d is shifted along itself by t_d, we can state that:

$$RAM_{I_T,i_T,j_T} = RAM_{I,i,j} \quad i \rightarrow i_T, j \rightarrow j_T \quad \forall i, j \in I \quad (7)$$

Using the same argument as stated in Section III-C3, RAM is unchanged since translation of axes also doesn't change the relative ordering of points. This is a useful property since shifting the origin to make the mean of the data equal to 0 is commonly used. Since such transformation is isometric, any distance-based metrics such as those in the Minkowski family are not affected by this transformation. However, metrics which measure some variation of angular distance, such as cosine similarity, result in different values when the origin is shifted.

5) Data set Awareness

When finding the proximity between two data points, existing proximity metrics do not utilize the information about the rest of the data set other than the two points of interest. This means that even though we can calculate the proximity by using the vector of the two points, it is not possible to contextualize that value. For datasets with dense clusters, average distances between any two data points is very small and hence identifying promixal points become very sensitive. This is because such metrics do not factor in the relative position of the rest of the data points. Also, modifying the underlying data set doesn't change the value of such proximity metrics as long as the points being compared are unchanged. RAM differs from such metrics since it inherently depends on the rank of the data points which itself is dependent on the rest of the dataset. Modifying the data set by adding or removing data points will modify the ranks and hence RAM accordingly.

6) Ranking Methods

RAM ranks data instances according to their numerical values in each dimension. However, two or more data points can have same values in a particular direction. This situation is more common for categorical or boolean features in a data set. In such cases, multiple strategies can be utilized for resolving ties when calculating ranks using $Rank_d$ including (1) *average* (2) *min* (3) *max* (4) *first* and (5) *dense*.

Ranking methods define how partitions are drawn for points with the same value along an axis. For instance, *dense* ranking strategy allows one partition between groups. *first* is the only ranking method that depends on the order the points were in the data making it sensitive to the initial ordering and shuffling of the data. Since it is also the only method that assigns different ranks to points even when they have the same value, it leads to poor performance when a significant number of data points are equal in any dimension. It is to be noted that in case of such ties, using any ranking method can lead to possible partitions perpendicular to any axis being less than the number of points. However, to be mathematically consistent, we always use the total count of points as the number of possible partitions as shown in Equation 2. This can break down the relationship between probability of two points occurring in same sub-space as discussed in Section III-A. Using an appropriate ranking method based on the nature of the data can improve the performance of *RAM*. The performance of *RAM* for each ranking method is shown in Section IV.

7) Measure of dissimilarity

Similar to some similarity measures, $RAM \in [0,1]$ (Cases when $RAM = 1$ and $RAM = 0$ are discussed in Section III-C2 and Section III-D respectively), $1 - RAM$ can be used as a measure of dissimilarity between two data points. However, even though $1 - RAM$ is always non-negative and commutative, it is not a distance function because $1 - RAM$ doesn't satisfy the triangle inequality (Equation 8).

$$RAM_{I,i,k} \nleq RAM_{I,i,j} + RAM_{I,j,k} \quad \forall i,j,k \in I \tag{8}$$

8) Computational Complexity

The computational complexity of the proposed measure to find the proximity matrix is $O(DN^2)$ where N^2 is the size of matrix that stores the proximity values for each possible pair of points. The measure requires the rank differences along all the D dimensions for every pair as shown in Equation 2. This complexity is identical to that of all the existing metrics for proximity search we have experimented with in this paper. For faster computation of Equation 2, we utilize an approximation trick described in the Section III-D.

9) Space Complexity

All proximity search algorithms, including *RAM*, have space complexity of $O(N^2)$ where N is the number of data points. However, as discussed in Section III-C5, Rank adjacency measure requires the rank matrix of the data to be calculated beforehand to avoid repeated calculations which is an NxD matrix. This is in addition to the memory required for calculating *RAM* for each pair of points. Hence, the peak memory usage of this measure tends to be significantly high and it needs various computational techniques for implementation in large datasets.

D. IMPLEMENTATION

From Equation 2, we can see that calculating *RAM* for any pair of points involves three major operations – (1) calculating rank differences across all D dimensions, (2) normalizing them using N and multiplying the results together and (3) taking the D^{th} root of the product. If we are only interested in the nearest neighbors of the points, calculating *RAM* values for each pair would be computationally inefficient. To prevent this, we propose an optimization trick that skips the last two operations for pairs which are far apart in all dimensions by considering only those pairs where the difference in ranks across at least one dimension is within a certain threshold. This threshold is termed as *depth* and is provided as a hyper-parameter to the *RAM* algorithm. Its value ranges from 1 to $N - 1$. For any two points with absolute rank difference $> depth$ in all dimensions, we assume *RAM* is approximately 0. Mathematically, it can be stated that:

$$|Rank_d(i) - Rank_d(j)| > depth \; \forall d \in D \implies RAM_{I,i,j} = 0 \tag{9}$$

The shaded area in Figure 3 denotes the area of valid points for which *RAM* would be calculated against point C when using $depth = 1$. This includes points which have a rank difference of 1 with C in either of the two dimensions. For all other points, *RAM* would approximated to be 0 w.r.t. C.

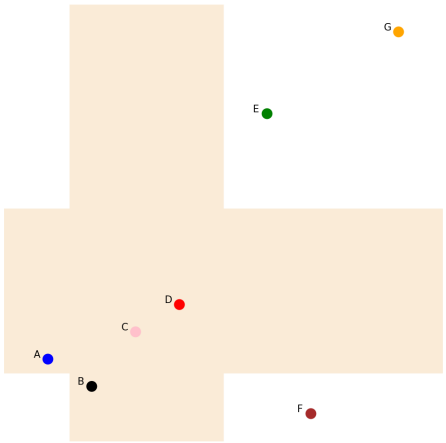

FIGURE 3: Valid points for calculating *RAM* w.r.t. C when $depth = 1$

High value of *depth* results in a more accurate rank adjacency matrix however it takes longer computation time. This relationship between *depth* and time is shown in Figure 4 where we experimented with various *depth* values using a sample of the MNIST data.

We also define a metric called *density* which is a function of *depth* and denotes the percentage of pairs in the data for which the actual *RAM* value was calculated. When $depth = N - 1$, the *RAM* would be calculated for all pairs and hence the density would be 100%. *density* allows us to empirically analyze the degree of difference in ordering of the points across all dimensions. If all dimensions have the same ranking of the

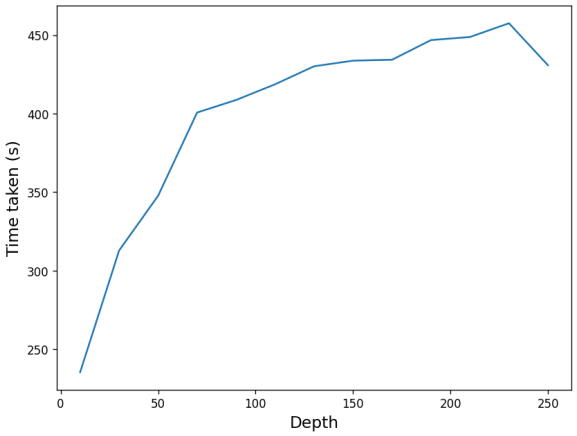

FIGURE 4: Sensitivity Analysis of Depth vs Time

points, rank difference being greater than depth in one dimension would mean it is higher than *depth* in all dimensions causing the actual rank adjacency to not be calculated. If the ranking in each dimension is different, it becomes more likely to find at least one dimension where the rank difference of two points is lower than *depth* leading to their rank adjacency to not be approximated. This effect can be visualized in Figure 5 where we started with a data with $N = 100$ and $D = 5$ where all the points were ordered similarly in each dimension (Noise scale = 0). Then we progressively added noise to the data to increase the differences in ordering across dimensions. Figure 5 shows that the *density* increases faster as the noise increases with the increase in *depth*.

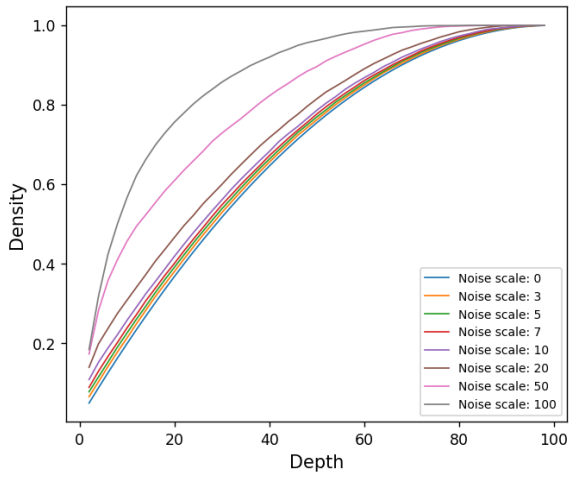

FIGURE 5: Sensitivity Analysis of Depth vs Density

Even though the exact value of density depends on the nature of the data, approximate upper bound and the optimal lower bound for density of the *RAM* matrix based on the data is shown in Equation 10 and Equation 11 respectively.

$$UB(density) = min(1, \frac{1 + (2 \text{ x } depth \text{ x } D)}{N}) \quad (10)$$

$$LB(density) = 1 - \frac{(N - depth - 1) \text{ x} (N - depth)}{N^2} \quad (11)$$

E. LIMITATIONS

In this section, we discuss certain limitations of the proposed method.

1) Orthogonality

RAM assumes orthogonality across dimensions and each dimension to be equally important. However, that is a strong assumption. Orthogonality can be induced via Principal Component Analysis (PCA), but it also induces importance disparity in the resulting dimensions. This causes most similarity measures to suffer in performance as shown in Section IV-B. However, *RAM* performs significantly worse than others after PCA transformation. One possible solution would be to scale the rank differences linearly based on the eigenvalues of the transformed dimensions in Equation 2. This will increase the effect of rank difference on dimensions that are more informative after PCA transformation while discounting the ones that are not.

2) Rotational Variance

RAM is also sensitive to any rotation of the data about any point including the origin, such as after PCA transformation, since rotation can change the rank ordering across dimensions.

3) Memory Complexity

For practical implementations, the memory requirements for *RAM* can be a bottleneck for large data sets. Since it is calculated within the context of a data set, the rank matrix, which is identical in size to the original data, needs to be computed and stored in memory. For massive data sets, the points can be binned across each axis and ranks can be calculated over the median of the bins for each axis. For high-dimensional data, it is unlikely for two points to lie in the same bins across all dimensions if the size of the data is much larger than the size of each bin and proximity values for most points would be unique even after binning. This approach trades small inaccuracies in proximity values for a major reduction in memory usage. If the bins span the entire range of values for each axis, the memory usage can be made static and independent of N. For static data sets, there are certain other ways to mitigate memory limitations. The rank matrix can be discarded once the *RAM* matrix has been populated. The proximity matrix can also be computed dimension by dimension reducing the peak memory usage at the cost of compute time. If the $depth << N$, the underlying data structure for *RAM* matrix can be switched to a sparse matrix for a significant deduction in memory since most values in the matrix would then be 0.

4) Dynamism

Dynamism is a major limitation of the proposed heuristic. The proximity measure relies on the rank matrix of the data which

would have to be modified in entirety every time the data is updated. There exists streaming algorithms for approximating ranks. However, they are impractical in terms of memory and compute as one such would have to be implemented for every dimension. One approach is to utilize the binning strategy defined in Section III-E3 wherein ranks are calculated over bins instead of data points. Any incoming data would be put in an appropriate bin across each axis to calculate their *RAM* values without any updates to the rank matrix. Another possible approximation can be to recalculate ranks for a small random subset of dimensions whenever a new data point is added. Then its nearest neighbor can be identified using *RAM* over those dimensions against a small random subset of points. The ranks for the new point over the remaining dimensions can then be approximated to be the same as that of its nearest neighbor.

IV. TEST CASES

To establish the effectiveness of *RAM*, we use three different test cases that use proximity search. We carefully select these test cases to include different types of data for performance validation. We use existing metrics for proximity search to compare the performance of RAM with existing methods. Here, we discuss the three test cases in detail in this Section.

All the experiments and models in this study are built on Ubuntu20 platform, Intel Xeon(R) processor with 256 GB RAM. We use Python 3.10 and open-source python libraries for all the analysis.

A. OUTLIER DETECTION

Outlier detection aims to identify data points that are significantly different from other points in the dataset. Many existing works use supervised learning methods for outlier detection. However, it is expensive to annotate data points as outliers (e.g. fraud transactions). Moreover, it is difficult to annotate if a given data point (or series of data points) reflects abnormal (outlier) behavior. In this test case, we use *RAM* to identify outliers in a dataset.

We have used the Credit Card Fraud Detection data set from Kaggle which contains $284,807$ transactions made by credit cards by European cardholders in September 2013. This data set is heavily imbalanced with 0.172% of transactions classified as fraud. This data set has 31 data fields. Apart from the target variable, this data contains 28 numerical features which are a result of a PCA transformation. No background knowledge is provided for these features. The remaining two columns are - *time* which denotes the number of seconds elapsed from the first transaction in the data and *amount* which denotes the transaction amount. Due to memory constraints, we have performed stratified random sampling to extract 25% of the data points for our experiments. The sampled data contains 71202 transactions with 123 i.e. 0.172% transactions marked as fraud.

To identify outliers using *RAM*, we calculate the rank adjacency of each data point till a certain depth. Then we calculate the average rank adjacency of its nearest k neighbors. Data points are sorted based on their top-k average rank adjacency and the ones with smallest values are classified as outliers. We hypothesize that outlier points may have high rank adjacency with each other but are likely to have poor rank adjacency with the rest of the data. For any point $i \in I$, its top-k average rank adjacency can be defined as:

$$\overline{RAM}_{I,i}(k) = avg(\text{top-}k\{RAM_{I,i,j} \forall j \in I\}) \tag{12}$$

To check the performance of RAM in outlier detection, we compare it against three commonly used unsupervised methods - Isolation Forest [19], Elliptic Envelope [20] and Local Outlier Factor [21] For all algorithms, the contamination ratio of 0.2% was used. For *RAM*, $depth = k = 2000$ and the *average* ranking method was used. Results from the experiments are shown in Table 1.

TABLE 1: Results of Outlier Detection

	Performance Values		
Method	Precision	Recall	f_1
RAM	**0.33**	**0.38**	**0.35**
Isolation Forest	0.22	0.26	0.24
Elliptic Envelope	0.09	0.11	0.10
Local Outlier Factor	0.01	0.01	0.01

Results in Table 1 show that RAM outperforms other commonly used methods for outlier detection by up to 50% across different metrics. Since all features in the dataset had distinct real values for each data point, *RAM* performs similarly for other ranking methods mentioned in Section III-C6. Next, we discuss Neighborhood search, an important problem in machine learning community.

B. NEAREST NEIGHBOR SEARCH

Nearest neighbor search or neighborhood search is a very common application of machine learning. It is used to find a data point that is closest to the data point of interest. Common applications include segmentation (clustering similar consumers for campaigning) or product recommendations (finding products with embeddings closest to the context embeddings). To test the performance of RAM in neighborhood search, we use two datasets – MNIST (a 30% stratified and shuffled sample containing 21000 images has been used due to memory constraints) for image [22] and IRIS as a small tabular data set [23]. To further assess how the performance of our method compares to that of existing ones under common data processing techniques, we have performed the following operations on the MNIST sample – (1) Std MNIST: Standardization with mean 0 and variance 1 to evaluate the effect of scaling and translation of axes (2) PCA(0.99): PCA with 99% explained variance to evaluate the effect of forced orthogonality and rotation of axes (3) VT(0): Variance threshold of 0 to remove any constant features.

To test the performance of the proposed method, we design the experiments differently from how these datasets are typically used in extant literature in Machine Learning. We evaluate the performance of each method by assessing how

many neighbors of each point have the same label as that point by making an assumption that its nearest neighbors are most likely to be points of the same class. Similarly, we assume that the furthest neighbors of any point are likely to have a different label as that point. Based on that idea, we have defined the following two evaluation metrics for this experiment – (1) Nearest Neighbor Accuracy with average percentage of the nearest k neighbors of all data points that have the same label as them and (2) Furthest Neighbor Accuracy with average percentage of the furthest k neighbors of all data points that do not have the same label as them.

Nearest neighbors identified from the MNIST data for a few samples are shown in Figure 6. The first column in the figure denotes the sample and the rest denote its nearest neighbors with their proximal neighbors, decreasing from left to right.

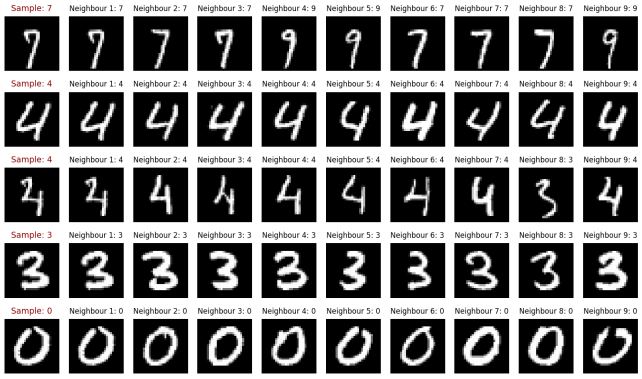

FIGURE 6: Nearest neighbors from MNIST using *RAM*

We test and compare the performance of *RAM* against six commonly used metrics. We use $k = 25$ for MNIST and $k = 10$ for Iris data to calculate nearest and furthest neighbor accuracy. For *RAM*, depth of 200 was used for the experiments with MNIST dataset. For the Iris dataset, full rank adjacency matrix was computed. Results for the nearest and furthest neighbor accuracy are shown in Table 2 and Table 3 respectively.

TABLE 2: Results of Nearest Neighbor Search

	MNIST				IRIS
Method	Original	Std	PCA	VT	Original
RAM_{avg}	89.3%	**89.3%**	53.0%	89.3%	92.1%
RAM_{min}	87.8%	87.8%	53.0%	87.8%	91.8%
RAM_{max}	85.6%	85.6%	53.0%	85.6%	92.0%
RAM_{first}	70.0%	70.0%	53.0%	71.8%	92.4%
RAM_{dense}	88.0%	88.0%	53.0%	88.0%	93.6%
Euclidean	89.3%	84.0%	84.1%	89.3%	93.9%
Manhattan	87.9%	86.4%	77.8%	87.9%	93.7%
Cosine	90.8%	84.2%	84.3%	90.8%	**94.2%**
Braycurtis	89.7%	88.0%	79.7%	89.7%	93.5%
Chebyshev	52.5%	56.0%	83.0%	52.5%	93.0%
Correlation	**90.9%**	84.6%	**84.3%**	**91.0%**	92.9%

Results show that barring PCA data, *RAM* performs as well as most existing algorithms with its accuracy always being within a few percentage points of the best algorithm in each category. Results also indicate that the performance of *RAM* is identical for the original and the standardized MNIST data set which is consistent with its invariant properties defined in Section III-C3 and III-C4. For the PCA data, there is no difference among the various ranking methods since this PCA transformation has eliminated ties within the data. For the case where all constant features are removed, all algorithms, with the exception of RAM_{first} and *correlation*, have identical scores to that of the original data. This is because even though the absence of constant features can modify values for most methods, the relative ordering of the scores is still maintained.

TABLE 3: Results of Furthest Neighbor Search

	MNIST				IRIS
Method	Original	Std	PCA	VT	Original
RAM_{avg}	97.6%	97.6%	89.4%	97.6%	99.4%
RAM_{min}	96.6%	96.6%	89.4%	96.6%	99.7%
RAM_{max}	97.1%	97.1%	89.4%	97.1%	99.0%
RAM_{first}	93.5%	93.5%	89.4%	93.8%	99.4%
RAM_{dense}	98.1%	**98.1%**	89.4%	98.1%	99.5%
Euclidean	98.1%	90.1%	90.1%	98.1%	99.8%
Manhattan	**98.2%**	92.7%	90.1%	**98.2%**	99.8%
Cosine	93.6%	97.4%	97.4%	93.6%	**100%**
Braycurtis	94.0%	97.7%	85.9%	94.0%	**100%**
Chebyshev	91.2%	90.0%	89.9%	91.2%	**100%**
Correlation	95.1%	97.4%	**97.4%**	95.4%	**100%**

Next we discuss an application of *RAM* in the area of semantic similarity.

C. TEXT SIMILARITY

Natural Language Processing (NLP) is one of the groundbreaking achievements of modern machine learning. It aims at understanding languages by converting text into a sequence of numbers (embeddings) which are used to represent sentences or words. Text similarity is useful in NLP as it used for multiple tasks like grouping semantically similar texts (clustering customer reviews by their sentiments) or extracting a similar text using vector retrieval from vector store. To test the performance of *RAM* in text, we use two datasets - STS Benchmark [24] and GloVe [25]. STS dataset is a standard data set for measuring the similarity between two sentences whereas GloVe is an open-source word embeddings dataset.

STS Benchmark data contains English sentences from news, forums and image captions for analyzing semantic textual similarity. It contains 5749 data points each containing a pair of sentences and an associated similarity score provided by human annotators which ranges from 0 (no at all similar) to 5 (extremely similar). We have used all 11498 sentences in STS to define the data set for *RAM* since it can only be applied within the context of a data set (as mentioned in Section III-C5). We use *all-MiniLM-L12-v2* embedding model to encode these sentences and then apply the proposed methodology to find the similarity for each pair of texts with *average* ranking method and $depth = 200$. The results for this experiment are shown in Figure 7 where the STS scores are on the x-axis and *RAM* dissimilarity $(1 - RAM)$ on the y-axis. We have used *RAM* dissimilarity so that the similarity values follow the same trend (value decreasing with increase in

proximity) as the existing comparison metrics. A correlation heat map of all methods and STS scores is shown in Figure 8. It can be seen that *RAM* corresponds closely to STS scores with a correlation value of 0.848.

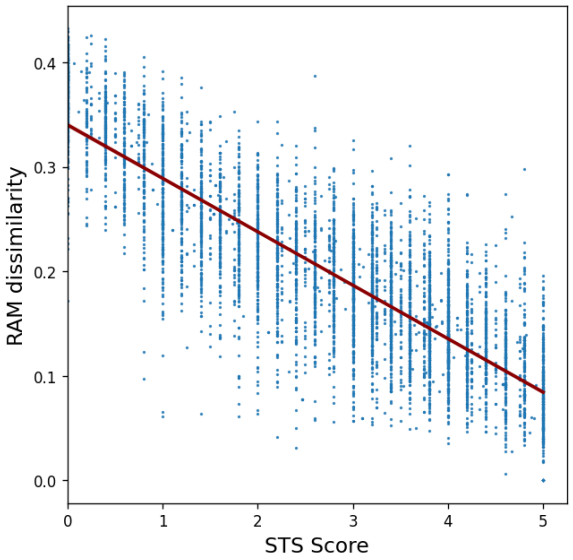

FIGURE 7: STS score vs RAM dissimilarity

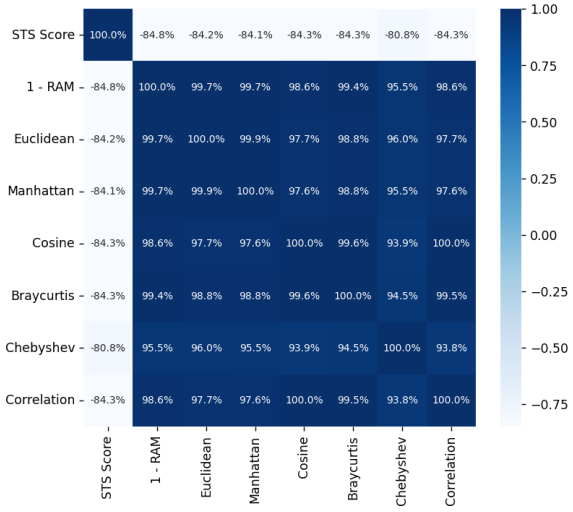

FIGURE 8: Correlation between all scores for STS

There are no existing standardized benchmarks for evaluating word similarity, so we rely on checking if *RAM* is able to retrieve similar words from the GloVe data which is a set of pre-trained word embeddings. We have used GloVe version with a vocabulary of $400K$ and contains 50-dimensional vectors for each word. We have used the first 20% ($80K$ words) of the vocabulary to limit ourselves to a more useful corpus and satisfy the memory constraints. For comparison, we have used cosine similarity since it is the most prominent method in text similarity. We pick four words at random and show the nearest ten words identified via both *RAM* and cosine similarity from the sampled GloVe data in Table 4, shown in order of decreasing similarity. The results shows that RAM is able to understand and retrieve semantically and contextually similar words. The set of words extracted by both methods also have a significant overlap.

V. CONCLUSION

Proximity search is a critical field of research for the advancements in machine learning applications. Existing methods consider distance based metrics which inherit multiple disadvantages including requirement of the knowledge of distribution and decreased performance as the number of dimensions increase. In search for improved alternatives, we introduce a novel rank-based measure for proximity search in this paper. We name this measure as Rank Adjacency Measure, or *RAM*. *RAM* uses an aggregate ranking approach across different dimensions to find proximity between any two data points/instances while considering the information contained in the rest of the dataset. It inherits properties from rank-based metrics e.g. robustness and insensitivity to large spaces between data points. Furthermore, it has multiple advantages. First, it is agnostic to the distribution of the data along multiple dimensions. Second, it can handle multi-dimensional numerical, categorical and nominal datasets. Third, it is intuitive and easy to implement for finding the similarity scores between data instances.

To evaluate the performance of the proposed *RAM*, we test *RAM* extensively across three test cases using five datasets. While our proposed measure outperforms the extant commonly used methods on outlier detection, it performs as well as others for neighborhood search and Text similarity. This research has certain limitations as discussed in the paper. Therefore, our work can be extended in multiple directions.

REFERENCES

[1] F. B. Insights, "The global machine learning (ml) market is expected to grow from $ 21.17 billion in 2022 to $ 209.91 billion by 2029n," https://www.fortunebusinessinsights.com/machine-learning-market-102226, accessed: 2023-10-11.

[2] J. Yu, Q. Tian, J. Amores, and N. Sebe, "Toward robust distance metric analysis for similarity estimation," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 1. IEEE, 2006, pp. 316–322.

[3] J. Li, J. Zhu, Q. Bi, G. Cai, L. Shang, Z. Dong, X. Jiang, and Q. Liu, "Miner: multi-interest matching network for news recommendation," in *Findings of the Association for Computational Linguistics: ACL 2022*, 2022, pp. 343–352.

[4] W. Shi, J. Michael, S. Gururangan, and L. Zettlemoyer, "Nearest neighbor zero-shot inference," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2022, pp. 3254–3265.

[5] P. Guo and W. Hu, "Potluck: Cross-application approximate deduplication for computation-intensive mobile applications," *SIGPLAN Not.*, vol. 53, no. 2, p. 271–284, mar 2018.

[6] N. Bhatia and Vandana, "Survey of nearest neighbor techniques," *arXiv preprint arXiv:1007.0085*, 2010.

[7] J. Lin, R. Pradeep, T. Teofili, and J. Xian, "Vector search with openai embeddings: Lucene is all you need," *arXiv preprint arXiv:2308.14963*, 2023.

[8] S.-H. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," *City*, vol. 1, no. 2, p. 1, 2007.

[9] L. R. Dice, "Measures of the amount of ecologic association between species," *Ecology*, vol. 26, no. 3, pp. 297–302, 1945.

TABLE 4: Similar Words in Text Similarity

	banks		austria		elbows		cisterns	
	RAM	cosine	RAM	cosine	RAM	cosine	RAM	cosine
1	investments	bank	switzerland	germany	rubbing	rubbing	cistern	cistern
2	depositors	lenders	belgium	switzerland	ankles	shoulders	fountains	aqueducts
3	lenders	credit	austria-hungary	sweden	patting	forearms	stove	aquifers
4	bank	loans	luxembourg	austrian	gloved	rubbed	baths	rainwater
5	withdraw	firms	poland	denmark	cuffed	hips	salts	canals
6	bailing	investments	hungary	belgium	finger	patting	canals	reservoirs
7	investment	investors	slovakia	luxembourg	rubbed	fingers	furnaces	jetties
8	brokerages	banking	denmark	hungary	buttocks	wrists	high-pressure	ditches
9	managed	funds	scandinavia	netherlands	legs	knees	reservoirs	culverts
10	citigroup	financial	museeuw	poland	cupped	ankles	greenhouses	baths

[10] J. C. Gower, "A general coefficient of similarity and some of its properties," *Biometrics*, pp. 857–871, 1971.
[11] R. Bellman and R. Kalaba, "On adaptive control processes," *IRE Transactions on Automatic Control*, vol. 4, no. 2, pp. 1–9, 1959.
[12] W. Krichene and S. Rendle, "On sampled metrics for item recommendation," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1748–1757.
[13] T.-H. Chiang, H.-Y. Lo, and S.-D. Lin, "A ranking-based knn approach for multi-label classification," in *Asian conference on machine learning*. PMLR, 2012, pp. 81–96.
[14] T.-Y. Liu, "Learning to rank for information retrieval," *Foundations and Trends in Information Retrieval*, vol. 3, no. 3, pp. 225–331, 2009.
[15] J. Jureckova, "Asymptotic linearity of a rank statistic in regression parameter," *The Annals of Mathematical Statistics*, vol. 40, no. 6, pp. 1889–1900, 1969.
[16] S. Chatterjee, "A new coefficient of correlation," *Journal of the American Statistical Association*, vol. 116, no. 536, pp. 2009–2022, 2021.
[17] S. C. Bagui, S. Bagui, K. Pal, and N. R. Pal, "Breast cancer detection using rank nearest neighbor classification rules," *Pattern Recognition*, vol. 36, no. 1, pp. 25–34, 2003.
[18] R. Fagin, R. Kumar, and D. Sivakumar, "Efficient similarity search and classification via rank aggregation," in *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, 2003, pp. 301–312.
[19] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 eighth ieee international conference on data mining*. IEEE, 2008, pp. 413–422.
[20] P. J. Rousseeuw and K. van Driessen, "A fast algorithm for the minimum covariance determinant estimator," *Technometrics*, vol. 41, no. 3, pp. 212–223, 1999.
[21] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 93–104.
[22] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.
[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
[24] J. O'shea, Z. Bandar, and K. Crockett, "A new benchmark dataset with production methodology for short text semantic similarity algorithms," *ACM Transactions on Speech and Language Processing (TSLP)*, vol. 10, no. 4, pp. 1–63, 2014.
[25] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.

MAYUR GARG is currently working in United Airlines, India as a Data Scientist. His work focuses on predicting airline fares and traffic of different consumer channels. He received his B.Tech from Maharaja Agrasen Institute of Technology, Delhi in 2019. From 2019 to 2022, he worked in Deloitte USI as NLP data scientist. From 2022 to 2024, he worked in Samsung Research Institute Bangalore, India as Senior Data Scientist. His reearch interest includes Data Science, NLP, Generative AI and Machine Learning.

DR ASHUTOSH NAYAK is currently working in Samsung Research Institute Bangalore in India in Ads Platform Data Intelligence Lab. His work focuses on recommender systems and Generative AI. He received his Bachelors and Masters in Technology from Indian Institute of Technology, Kharagpur India in 2014. From 2014 to 2018, he was a PhD student in Industrial Engineering at Purdue University. He worked in Dynamic Optimization for Smart Grid. From 2019 to 2022, he worked as a postdoc in Graduate School of Management at University of California, Davis. His work focused on understanding consumer behavior in mobile apps. His research interest includes static and dynamic optimization, recommender systems and Generative AI.

DR. D M RAJA SEKHARA REDDY Received the Ph.D., from University of Fribourg, Switzerland, in 2007. He is currently working Samsung Research Institute Bangalore in India in Ads Platform Data Intelligence Lab. His current research interests focus on development of scalable ML models using big data across Samsung services with special emphasis on user engagement management using Conversational Agents and personalized product recommendations for Direct-2-Customer(D2C) business.